

# Large-Scale Dense 3D Reconstruction from Stereo Imagery

Pablo F. Alcantarilla, Chris Beall and Frank Dellaert

**Abstract**—In this paper we propose a novel method for large-scale dense 3D reconstruction from stereo imagery. Assuming that stereo camera calibration and camera motion are known, our method is able to reconstruct accurately dense 3D models of urban environments in the form of point clouds. We take advantage of recent stereo matching techniques that are able to build dense and accurate disparity maps from two rectified images. Then, we fuse the information from multiple disparity maps into a global model by using an efficient data association technique that takes into account stereo uncertainty and performs geometric and photometric consistency validation in a multi-view setup. Finally, we use efficient voxel grid filtering techniques to deal with storage requirements in large-scale environments. In addition, our method automatically discards possible moving obstacles in the scene. We show experimental results on real video large-scale sequences and compare our approach with respect to other state-of-the-art methods such as *PMVS* and *StereoScan*.

## I. INTRODUCTION

Structure from Motion (SfM) and visual Simultaneous Localization and Mapping (vSLAM) algorithms [1, 11] aim to recover a sparse 3D reconstruction and the estimated camera poses in large-scale environments. These methods track features between different frames and optimize 3D structure and camera poses in a nonlinear optimization which incorporates the geometric multi-view constraints between 3D structure, camera poses and image measurements. This nonlinear optimization problem is normally solved by using bundle adjustment variants [10].

Sparse 3D models do not provide enough detail to fully appreciate the underlying structure of the environment. To this end, there have been various efforts towards automated dense 3D reconstruction in the last few years [8, 14, 3, 4, 13, 6]. Automated dense 3D modeling facilitates scene understanding and has countless applications in different areas such as augmented reality, cultural heritage preservation, autonomous vehicles and robotics in general.

One of the key ingredients in dense 3D reconstruction methods is *Multi-View Stereo* (MVS) [16]. MVS algorithms can be roughly classified into four different categories: *deformable polygonal meshes* [2], requiring a visual hull model as an initialization; *voxel-based* [13], requiring a bounding box that contains the scene and the accuracy is limited by the voxel grid size; *patch-based* [3], requires reconstruction of a collection of multiple small surface patches, and *multiple depth maps* [8, 14, 6], that demands fusing multiple maps into a single global model. As mentioned in [3], MVS algorithms can also be thought of in terms of the datasets they

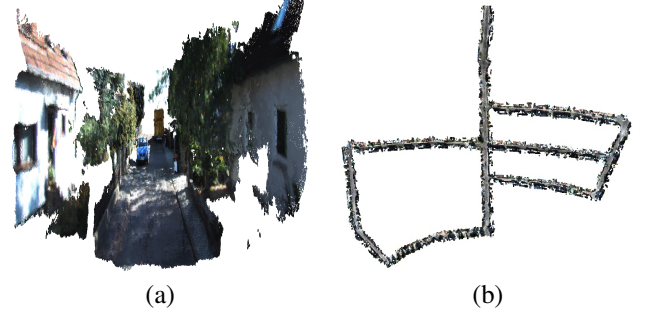


Fig. 1. Details (a) and aerial view (b) of dense 3D reconstruction results for a sequence of 2.2 Km and 2760 frames. The number of reconstructed 3D points is 5,770,704.

can handle: a single object, large-scale scenarios, crowded environments, etc. The choice of a particular MVS algorithm highly depends on the type of dataset and application of interest.

In this paper, we are interested in dense 3D reconstruction of large-scale environments using stereo imagery from a moving platform. We focus on the scenario of a stereo camera mounted on a vehicle or a robot exploring a large scene such as the one depicted in Figure 1. Large-scale environments pose new challenges to the dense 3D reconstruction problem such as large storage requirements and computational complexity.

We propose a novel MVS approach that efficiently combines the best of previous MVS approaches for our target application. Instead of fusing raw disparity maps from each stereo frame (which invariably yields large storage requirements), we use the dense disparity maps as an initialization for a patch-based surface reconstruction considering multiple views. In this way, taking advantage of the flexibility of patch-based methods, we can check for geometric and photometric consistency of each individual patch, which facilitates discarding moving objects from the final reconstruction. Then, we use efficient voxel grid filtering to down-sample the dense point cloud for dealing with large storage requirements.

Our algorithm makes the assumption that the stereo rig calibration and camera motion are already known. Stereo calibration can be obtained offline, while camera motion can be obtained either online by incremental egomotion estimation methods such as visual odometry [6] or with an offline bundle adjustment optimization including loop closure constraints. Our algorithm has the following advantages:

- Exploits dense disparity maps using efficient stereo matching.

<sup>3</sup> The authors are with the School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA. {pfa3, cbeall13, frank}@cc.gatech.edu

- Performs efficient data association, checking for geometric and photometric consistency in a multi-view setup taking into account the uncertainty of stereo measurements.
- Handles large storage requirements due to the use of voxel grid filtering techniques.
- Is able to reject outliers and moving objects or obstacles in the scene.
- It is faster than state-of-the-art techniques.

## II. RELATED WORK

One of the most popular MVS techniques is the patch-based approach also known as PMVS [3]. This method builds a dense 3D reconstruction of a scene based on collections of multiple small surface patches. PMVS basically consists of three different steps: feature matching, expansion and filtering. In the matching step, a sparse 3D reconstruction of the scene is obtained from a set of 2D features. Then, in the expansion step, this sparse 3D point cloud is densified by an iterative procedure that estimates patch geometry by minimizing a photometric cost function. Finally, outliers are removed in the filtering step. PMVS is able to handle moving objects thanks to the photometric consistency check between different images. The main limitation of PMVS is that it is computationally very expensive due mainly to the patch expansion step. For large-scale scenarios, such as the ones we are interested in, PMVS would require several days to obtain dense 3D reconstructions even when using efficient clustering techniques for the set of input images [4].

Pollefeys *et al.* [14] presented an efficient approach for real-time 3D reconstruction from video of urban scenes. Their approach considers a system equipped with 8 cameras plus GPS/INS data mounted on a moving car, exploiting parallelization and GPU processing. They use plane-sweeping stereo as a stereo matcher for obtaining dense disparity maps from different views. Then, multiple depth maps are fused into a single global model by exploiting visibility information.

Recently, Newcombe *et al.* presented an impressive voxel-based dense 3D reconstruction approach from monocular imagery [13]. This approach works well for small scale environments and requires prior knowledge for a bounding box that contains the scene, limiting the accuracy of the 3D reconstruction to the voxel grid resolution.

The approach most similar to ours is the *StereoScan* system described in [6]. In this approach, the authors propose a dense 3D reconstruction pipeline fusing information from dense disparity maps obtained from stereo imagery. In order to deal with the large amount of data from the fusion of multiple disparity maps, the authors propose a greedy approach for solving the data association problem between two consecutive stereo frames. This greedy approach simply reprojects reconstructed 3D points of the previous frame into the image plane of the current frame. When a point projects to a valid disparity, the 3D points from the current and previous frames are fused by computing their 3D mean. Similar to our approach, the authors assume that the camera motion

is obtained from an independent visual odometry pipeline working in parallel. The main limitation of StereoScan is its greedy data association approach that considers only two consecutive frames without checking for geometric and photometric consistency between the reconstructed points. Limiting the data association to just two frames and without checking for geometric and photometric consistency introduces many noisy points into the final model, without being able to deal with possible artifacts caused by dynamic objects that will corrupt the 3D model. In addition, without filtering, the storage requirements quickly become prohibitive for large-scale scenarios.

## III. STEREO VISION

Stereo vision makes it possible to estimate 3D scene geometry given only two images from the same scene. We consider a conventional stereo rig in which two cameras are separated by a horizontal baseline. Rectification [9] considerably simplifies the stereo correspondence problem and allows for straight-forward computation of dense disparity maps, which form the base for the dense 3D reconstruction. Each value in the disparity map can be reprojected to a 3D point  $h_i = (x, y, z)^t \in \mathbb{R}^3$  with respect to the camera coordinate frame based on the projective camera equations:

$$\begin{aligned} z &= f \cdot \frac{B}{u_R - u_L} = f \cdot \frac{B}{d_u} \\ x &= \frac{z \cdot (u_L - u_0)}{f} \\ y &= \frac{z \cdot (v - v_0)}{f} \end{aligned} \quad (1)$$

where  $f$  is the camera focal length,  $(u_0, v_0)$  is the principal point,  $B$  is the stereo baseline and  $(u_L, v_L)$  and  $(u_R, v_R)$  are the stereo measurements in the left and right images, respectively. Note that for rectified stereo images  $v_L = v_R = v$ . The horizontal disparity  $d_u$  is the difference in pixels between the horizontal image projections of the same 3D point in the right and left images.

Similarly to [12], our sensor error model is composed of two parts: *pointing error*  $\sigma_p$  and *matching error*  $\sigma_m$ . Pointing error is the error in image measurements due to camera calibration inaccuracy, whereas matching error is due to the inaccuracy of the stereo matching algorithm. Given these values, we can compute the covariance matrix of the stereo measurements  $(u_L, v, d_u)^t$  in the disparity space as:

$$S_i = \begin{bmatrix} \sigma_p & 0 & 0 \\ 0 & \sigma_p & 0 \\ 0 & 0 & \sigma_m \end{bmatrix} \quad (2)$$

To obtain the covariance matrix  $P_i$  of the reconstructed 3D point  $h_i$  associated with stereo measurements  $(u_L, v, d_u)^t$ , the error is propagated from the 2D measurement space to 3D by means of linear uncertainty propagation as:

$$P_i = J_i \cdot S_i \cdot J_i^t \quad (3)$$

$$J_i = \begin{bmatrix} \frac{\partial x}{\partial u_L} & \frac{\partial x}{\partial v} & \frac{\partial x}{\partial d_u} \\ \frac{\partial y}{\partial u_L} & \frac{\partial y}{\partial v} & \frac{\partial y}{\partial d_u} \\ \frac{\partial z}{\partial u_L} & \frac{\partial z}{\partial v} & \frac{\partial z}{\partial d_u} \end{bmatrix} = \begin{bmatrix} \frac{B}{d_u} & 0 & -\frac{u_L B}{d_u^2} \\ 0 & \frac{B}{d_u} & -\frac{v B}{d_u^2} \\ 0 & 0 & -\frac{f B}{d_u^2} \end{bmatrix} \quad (4)$$

where  $J_i$  is the Jacobian of the 3D point  $h_i$  with respect to the stereo measurements  $(u_L, v, d_u)^t$ . The covariance matrix  $P_i$  estimates the uncertainty we can expect from a reconstructed 3D point. The uncertainty error grows quadratically with respect to the depth. We denote by  $w_i = \text{trace}(P_i)$  the trace of the covariance matrix  $P_i$ , and this is used as a measure of the uncertainty and as a weighting function for the reconstructed 3D points and color information in our MVS approach.

#### IV. DENSE 3D RECONSTRUCTION

Our approach assumes that stereo camera calibration and motion are known. In addition, we assume that images are given in a time-ordered sequence. Our approach is applicable in batch as well as incremental modes. Camera motion can be obtained online by egomotion estimation methods such as visual odometry or after an offline bundle adjustment optimization including possible loop closure constraints.

Our dense reconstruction approach has these main steps:

- 1) Dense stereo matching.
- 2) Patch-based reconstruction with multi-view geometric and photometric consistency analysis.
- 3) Outlier removal and voxel grid filtering.

We first select a subset of stereo keyframes from the input images to enforce a minimum distance in camera motion between frames which will be processed. This is to avoid adding redundant images which would not contribute any new information to the dense 3D model, but only increase computational complexity. Each stereo keyframe  $F_k$  with  $k = 1 \dots N$  comprises:

- Camera rotation,  $R^k \in SO(3)$ .
- Camera translation,  $\mathbf{t}^k \in \mathbb{R}^3$ .
- Left rectified RGB image,  $I_L^k: \mathbb{R}^2 \rightarrow \mathbb{R}^3$ .
- Normalized zero mean and unit variance left rectified RGB image,  $I_{Lnorm}^k: \mathbb{R}^2 \rightarrow \mathbb{R}^3$ .
- Right rectified RGB image,  $I_R^k: \mathbb{R}^2 \rightarrow \mathbb{R}^3$ .
- Disparity map,  $I_D^k: \mathbb{R}^2 \rightarrow \mathbb{R}$ .

The camera rotation and translation are defined such that a 3D point  $Y_i = (x, y, z)^t \in \mathbb{R}^3$  in the world coordinate frame can be transformed into the camera coordinate frame with:

$$h_i = R^k (Y_i - \mathbf{t}^k) \quad (5)$$

and assuming a pin-hole camera model, the projection of the 3D point  $h_i$  into the image plane is:

$$U_i = K (R^k (Y_i - \mathbf{t}^k)) \quad (6)$$

where  $K$  is the matrix representing the camera intrinsics and  $U_i = (u, v, 1)^t$  is the vector of pixel measurements in homogeneous coordinates. In addition, each 3D point  $h_i$  has

an associated RGB color vector  $c_i = (r, g, b)^t \in \mathbb{R}^3$ . Now, we will describe in detail each of the main steps in our MVS algorithm.

##### A. Dense Stereo Matching

Reliable stereo matching is critical in order to obtain accurate dense 3D point clouds. For this purpose, we use the Efficient Large-Scale Stereo Matching (ELAS) method which is freely available [5]. ELAS provides dense high quality disparity maps without global optimization, while remaining faster than many other stereo methods. For each stereo keyframe  $F_k$  we obtain a dense disparity map  $I_{Disp}^k$  image from the left and right rectified images.

##### B. Multi-View Geometric and Photometric Consistency

Considering that each stereo frame gives rise to thousands of 3D points, transforming all of these into a global 3D model would yield a very noisy reconstruction with lots of redundant points, and consequently storage requirements of prohibitive proportions for large scenarios. Therefore, to avoid the introduction of many redundant points we solve the data association problem between multiple stereo frames and verify geometric and photometric consistency for all points. This is in principle similar to the photometric consistency employed in MVS approaches [8, 3] with the key difference that for each pixel we rely on the depth provided by the stereo matching algorithm instead of minimizing a photometric cost function to find the globally optimal depth of each patch. Figure 2 depicts a graphical example of our multi-view stereo approach considering three views.

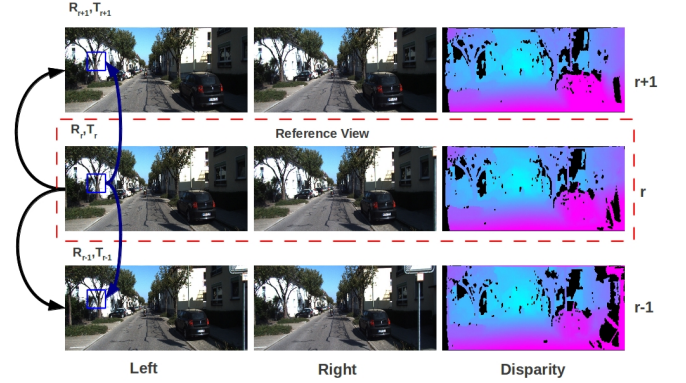


Fig. 2. Multi-view stereo approach checking geometric and photometric consistency. For a pixel  $p$  in the left reference image that has a valid disparity, we check first for geometric consistency between the different views. If the geometric consistency is successful, we perform the photometric consistency analysis.

We choose a central reference stereo keyframe  $F_r$  in a local neighborhood of  $m$  stereo views (in our experiments we consider  $m = 3, 5$ ). The index of the reference stereo keyframe  $r$  in the local neighborhood of  $m$  stereo frames is taken as the central view,  $r = (m - 1) / 2 + 1$ .

For each pixel  $p = (u_L, v_L)$  from the left reference keyframe image  $I_L^r$  which has a valid disparity  $d_u$ , we first perform a geometric consistency check with respect to the

other views in the neighborhood. We compute the 3D point  $h_i$  and the associated covariance  $P_i$  as described in Eq. 1 and Eq. 3 respectively. If the trace of the covariance matrix  $w_i$  is below some threshold  $T_{cov}$ , we then project the point  $h_i$  into the left images for the other  $m - 1$  views in the neighborhood. We then check that the projection of each 3D point  $h_i$  from the reference view into neighboring frames has a valid disparity and low uncertainty. Finally, we also check that the 3D difference between all reconstructed 3D points expressed in the world coordinate frame is within a threshold  $T_{dist}$ .

For all 3D points from the reference image which passed the geometric consistency check, our algorithm then proceeds to a photometric consistency check with respect to the other views in the neighborhood. For each pixel  $p = (u_L, v_L)$  from the left reference image, we compute the normalized cross correlation  $NCC(F_r, F_k, p)$  between a  $\mu \times \mu$  window centered on  $p$  and the corresponding windows centered on the projections in each of the views  $F_k$  with subpixel accuracy. For the  $NCC$  we use the textures from the normalized zero mean unit variance left images  $I_{Lnorm}^k$ . Similar to [8] we use a version of  $NCC$  for  $l$ -dimensional RGB color vectors with normalization per color channel.

$$NCC(c_0, c_1) = \frac{\sum_{j=0}^{l-1} (c_0(j) - \bar{c}_0) \cdot (c_1(j) - \bar{c}_1)}{\sqrt{\sum_{j=0}^{l-1} (c_0(j) - \bar{c}_0)^2 \cdot \sum_{j=0}^{l-1} (c_1(j) - \bar{c}_1)^2}} \quad (7)$$

The  $NCC$  returns a scalar value between  $[-1, 1]$ , where 1 indicates perfect correlation. We compute an average photometric score  $g(p)$  that comprises the sum of photometric scores for the pixel  $p$  between the reference image and the rest of views  $F_k \in V$  where the point is visible:

$$g(p) = \frac{1}{|V|} \sum_{k=r-n}^{k=r+n} NCC(F_r, F_k, p) \quad (8)$$

where  $n = (m - 1)/2$  for the sake of brevity, and  $|V|$  denotes the number of views where the point  $p$  is predicted to be visible, i.e. the number of views for which the point passed the geometric consistency check. If the mean photometric score  $g_p$  exceeds a threshold value  $T_{photo}$  and  $|V|$  is 3 or greater, we proceed to fuse the 3D point with respect to the world coordinate frame and color information into the dense reconstruction as the following weighted average:

$$Y_i = \frac{\sum_{k=r-n}^{k=r+n} w_{i,k} \cdot Y_{i,k}}{\sum_{k=r-n}^{k=r+n} w_{i,k}}, \quad c_i = \frac{\sum_{k=r-n}^{k=r+n} w_{i,k} \cdot c_{i,k}}{\sum_{k=r-n}^{k=r+n} w_{i,k}} \quad (9)$$

where  $w_{i,k}$  is the uncertainty weight of the reconstruction of point  $h_i$  from the view  $k$ . Similarly,  $Y_{i,k}$  and  $c_{i,k}$  denote the 3D point with respect to the world coordinate frame and color information for point  $i$  from view  $k$ .

In order to reduce computational complexity and to avoid adding redundant 3D points as the neighborhood window

slides through the sequence, we keep track of image projections of already reconstructed 3D points in their respective images using a mask. In this way, for each new reference view, we check the visibility masks to reconstruct only those 3D points which were not reconstructed previously.

### C. Outliers Removal and Voxel Grid Filtering

Once we have computed a dense 3D point cloud from a reference stereo keyframe  $F_r$ , we filter possible outliers by means of a *radius removal* filter. This filter removes those 3D points that do not have at least some number of neighbors within a certain range. Then, in order to reduce the computational burden and storage requirements, we downsample the 3D point cloud using a voxel grid filter that fits to the dimensions of the input point cloud. In each voxel, the 3D points are approximated with their centroid, representing more accurately the underlying surface. Once we have processed one stereo keyframe, we repeat the same procedure for the next keyframe until the sequence finishes. After processing all stereo keyframes, we apply the voxel grid filter over the whole dense 3D point cloud to fuse the 3D points into a global voxel grid structure.

## V. RESULTS

We use the KITTI visual odometry RGB dataset [7] for the evaluation of our dense 3D reconstruction approach. This dataset consists of stereo imagery with accurate stereo calibration. The images have a resolution of  $1241 \times 376$  pixels. For the greedy projection surface reconstruction and the radius removal and voxel grid filters, we use the efficient implementations from the Point Cloud Library (PCL) [15].

Typical values for the parameters in our method are:  $\sigma_p = 0.5$  pixels,  $\sigma_m = 1.0$  pixel,  $T_{cov} = 0.5$ ,  $T_{dist} = 0.5$  m,  $T_{photo} = 0.7$  and patch size  $7 \times 7$  pixels. All timing results were obtained with an Intel Core i7-3770 CPU.

### A. Comparison to PMVS and StereoScan

We compare our dense 3D reconstruction approach to PMVS and StereoScan. For PMVS we use the PMVS2 implementation<sup>1</sup>. We configure PMVS options so that it processes images in sequence, enforcing the algorithm to use only images with nearby indices to reconstruct 3D points. For the StereoScan case we use our own implementation and fuse the information between two corresponding 3D points if both disparities are valid and the distance between reconstructed 3D points is below the threshold  $T_{dist}$ . In our method we consider  $m = 3$  views, a voxel grid resolution of 5 cm and a photometric consistency threshold  $T_{photo} = 0.7$ . This value is also used in PMVS.

Figure 3(a) depicts a comparison of our method to PMVS and StereoScan showing the computation time versus the number of input images for the first sequence in the KITTI dataset. We observe that our method is the fastest one. The reason why it is faster than StereoScan is due to the use of a visibility mask, keeping track of image projections of the reconstructed 3D points in their visible images, reducing

<sup>1</sup>Available from: <http://www.dl.ens.fr/pmvs/>



computational complexity. PMVS is highly time consuming even for a small set of images. This is because it tries to optimize the 3D position and normal of each patch in each reference image by minimizing a cost function based on the photometric error in a multi-view setup. In contrast, our method and StereoScan use the available 3D geometry from the disparity map and perform data association between different views, which is faster than running an iterative non-linear optimization per patch.

Figure 3(b) shows a comparison of our method to PMVS and StereoScan showing the number of reconstructed 3D points versus the number of input images. The number of reconstructed 3D points in the StereoScan case was scaled down by a factor of ten for clarity reasons. StereoScan produces large amount of 3D points, some of which are noisy and redundant. In large-scale environments the storage requirements of StereoScan can become prohibitive. In contrast, our method returns a more reasonable number of 3D points. In addition, one can control the output number of 3D points with the photometric threshold and the voxel grid resolution. PMVS returns the lowest number of reconstructed 3D points. PMVS is more targeted to *Photosynth-type* systems [1], where there is a large number of images from the same object in a small area. In this case, redundant viewpoints improve the estimation of the patch geometry.

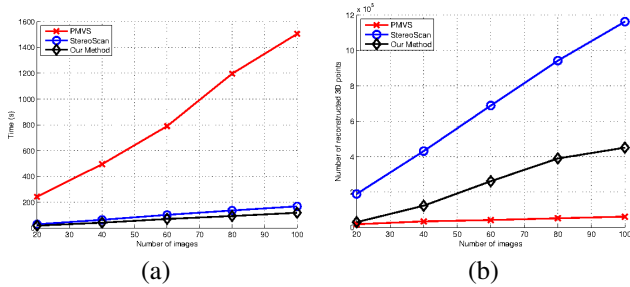


Fig. 3. Comparison to PMVS and StereoScan: (a) Computational time vs number of images (b) Number of reconstructed 3D points vs number of images. Note that the number of reconstructed 3D points that is reported for StereoScan is scaled by a factor of ten for clarity reasons.

Table I shows information about the number of reconstructed 3D points at each level of our MVS approach considering two different photometric thresholds  $T_{photo} = 0.2$  and  $T_{photo} = 0.8$ . In addition, we also show the percentage between the number of accepted points at each step and the number of points that have a valid disparity for each stereo frame. We can observe that in both cases the number of 3D points obtained after the voxel grid filtering is a small fraction of the original number of points facilitating storage requirements in large-scale scenarios.

### B. Detection of Moving Objects

One of the nice properties of PMVS and similar patch-based methods such as ours, is that they can discard specular highlights or moving objects in the scene (pedestrians, cars, etc.). Assuming that the surface of an object is Lambertian, the photometric score function  $g(p)$  will give low scores for

Step	Our method $T_{photo} = 0.2$	Our method $T_{photo} = 0.8$
# Points Disparity	323,420	323,420
# Points Geometric	133,334	133,334
% Accepted	41.23	41.23
# Points Photometric	57,675	9,310
% Accepted	17.83	2.88
# Points Fusion	8,851	1,885
% Accepted	2.74	0.58

TABLE I  
AVERAGE NUMBER OF RECONSTRUCTED 3D POINTS PER STEP AND PERCENTAGE OF ACCEPTED POINTS WITH RESPECT TO POINTS WITH VALID DISPARITY PER STEREO FRAME.

areas which have specular highlights or moving objects in the image, and therefore these points will not be added to the final 3D model. Figure 4 depicts an example of one sequence where there are several moving objects (cars). StereoScan fails to reject these points and adds them to the final model, creating artifacts in the final model. This occurs because StereoScan only considers two consecutive stereo frames for data association based on the disparity information. In such a limited multi-view setup moving objects are not detected properly. In contrast, our method and PMVS are able to discard those 3D points from the final model.



Fig. 4. Detection of moving objects. Top: Two frames from a sequence where there are moving objects in the scene. Bottom left: view of the dense 3D reconstruction with our method. Bottom right: view of the dense 3D reconstruction with StereoScan. Notice how artifacts due to the moving objects are introduced in the final model.

### C. 3D Reconstruction Results

Figure 5 depicts some dense 3D large-scale reconstruction results from different viewpoints. It can be noticed that the dense 3D point clouds contain high level of detail, enough for visualization purposes.

### D. Timing Evaluation

Table II shows average timing results for the most important operations in our MVS approach. We can observe that on average obtaining one incremental update to the dense



Fig. 5. Details of large-scale dense 3D reconstruction results. The cars in the point clouds correspond to static objects in the environment.

3D point cloud takes slightly less than 2 seconds for one stereo view. This time could be further reduced by using GPU implementations since the operations in the multi-view 3D reconstruction approach are independent per pixel.

Step	Time (ms)
Stereo Matching	157.74
RGB Normalization	2.51
Multi-view 3D (m=3)	1303.28
Outlier Removal	351.32
Voxel Grid Filter	2.76
<b>Total</b>	<b>1811.61</b>

TABLE II  
COMPUTATION TIMES IN MS FOR THE MAIN STEPS OF OUR MVS  
APPROACH.

## VI. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a novel MVS approach for dense 3D reconstruction in large-scale environments using stereo imagery. We have shown that efficiently fusing disparity maps, while checking geometric and photometric consistency of patches in a multi-view setup, yields detailed 3D models with low storage requirements. In the future we are interested in possible applications of the dense 3D models for planning and scene understanding.

## REFERENCES

- [1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. In *Intl. Conf. on Computer Vision (ICCV)*, 2009.
- [2] Y. Furukawa and J. Ponce. Carved visual hulls for image-based modeling. *Intl. J. of Computer Vision*, 81(1):53–67, 2009.
- [3] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. Pattern Anal. Machine Intell.*, 32(8):1362–1376, 2010.
- [4] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1434–1441, 2010.
- [5] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Asian Conf. on Computer Vision (ACCV)*, pages 25–38, 2010.
- [6] A. Geiger, J. Ziegler, and C. Stiller. StereoScan: Dense 3d reconstruction in real-time. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 963–968, 2011.
- [7] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012.
- [8] M. Goesele, B. Curless, and S. M. Seitz. Multi-view stereo revisited. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2402–2409, 2006.
- [9] R. Hartley. Theory and practice of projective rectification. *Intl. J. of Computer Vision*, 35:115–127, 1999.
- [10] Y. Jeong, D. Nistér, D. Steedly, R. Szeliski, and I. S. Kweon. Pushing the envelope of modern methods for bundle adjustment. *IEEE Trans. Pattern Anal. Machine Intell.*, 34(8):1605–1617, 2012.
- [11] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. REAL: A system for large-scale mapping in constant-time using stereo. *Intl. J. of Computer Vision*, 94(2):198–214, 2011.
- [12] D. R. Murray and J. J. Little. Environment modeling with stereo vision. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 3116–3122, 2004.
- [13] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *Intl. Conf. on Computer Vision (ICCV)*, pages 2320–2327, 2011.
- [14] M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénus, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3D reconstruction from video. *Intl. J. of Computer Vision*, 78(2-3):143–167, 2008.
- [15] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.
- [16] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. S. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 519–528, 2006.